

A Novel Weight-Shared Multi-Stage Network Architecture of CNNs for Scale Invariance

Ryo Takahashi, Takashi Matsubara, *Member, IEEE*, and Kuniaki Uehara, *Member, IEEE*

Abstract—Convolutional neural networks (CNNs) have demonstrated remarkable results in image classification tasks for benchmark and practical uses. The CNNs with deeper architectures have achieved higher performances recently thanks to their robustness to parallel shift of objects in images as well as their numerous parameters and resulting high expression ability. However, the CNNs have a limited robustness to other geometric transformations such as scaling and rotation. This problem is considered to limit performance improvement of the deep CNNs but there is no established solution. This study focuses on scale transformation and proposes a novel network architecture called *weight-shared multi-stage network* (WSMS-Net), consisting of multiple stages of CNNs. The WSMS-Net is easily combined with existing deep CNNs, such as ResNet and DenseNet, and enables them to acquire a robustness to scaling of objects. The experimental results demonstrate that existing deep CNNs combined with the proposed WSMS-Net achieve higher accuracy for image classification tasks only with a little increase in the number of parameters.

Index Terms—Image Classification, Scale Invariance, Convolutional Neural Network, Shared Weights

I. INTRODUCTION

Convolutional neural networks (CNNs) [13] have made great achievements in the tasks of image classification and image processing. They have already been employed for practical use in various situations. The CNNs are known to be robust to small parallel shift thanks to their architecture: Units in a CNN have their own local receptive fields, share weight parameters with other units, and are sandwiched by the pooling layers. The performance of the CNNs has been improved by development of new network architectures year after year. A brief history is found in the results of ImageNet Large Scale Visual Recognition Competition (ILSVRC), which is a competition of image processing by machine learning. After an 8-layer CNN AlexNet [11] recorded the highest accuracy in ILSVRC in 2012, a 19-layer CNN VGG [19] and a 22-layer CNN GoogLeNet [20] updated the state-of-the-art results in 2014 by deepening the number of layers, increasing the number of internal weight parameters, and thereby expanding the expression ability of the network. A further deep neural network model, ResNet [22], recorded the highest accuracy in ILSVRC in 2015. This new model has more than 100 convolution layers and has new shortcut connections that do not perform convolution. This architecture propagates the gradient through the shortcut connections and overcomes the gradient

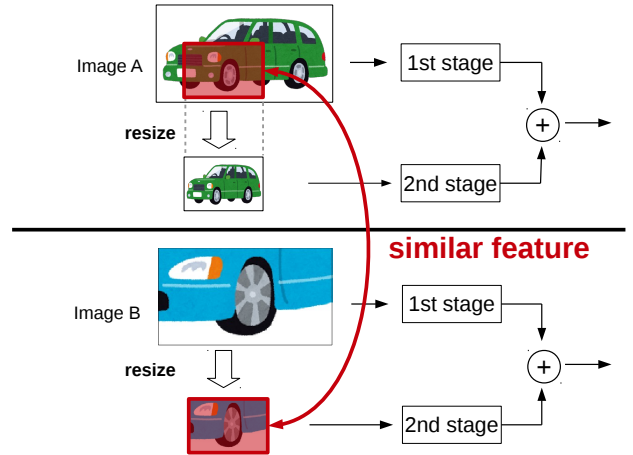


Fig. 1. Conceptual explanation of scale invariance in the WSMS-Net. The image A provides local features of an automobile such as the shapes of tires, windows, headlights, and license plate in the first stage. Similar features are detected in the resized image B, a part of sports car, in the second stage thanks to the shared weight parameters.

vanishing problem [18], which is considered as an obstacle to accuracy improvement of the deep neural networks. In 2016, the model based on ResNet also broke the record. Many studies have focused on devising the structure of the CNNs to prevent gradient vanishing problem (e.g., DenseNet [6]).

On the other hand, the fact that the CNN still has a limited robustness to geometric transformations other than parallel shift has not attracted enormous attention. The notable geometrical transformations of objects in image are scaling and rotation in addition to parallel shift [12]. Scaled and rotated objects in images are not recognized correctly or at least require parameters in addition to parameters for recognizing the original objects, resulting in a limitation to the expression ability of the CNNs. Several previous studies tried to acquire the invariance by combining with geometric transformation layers [15], [2]. These layers require localization of a object, a repeated computation for a single image, or computation between all-to-all pairs of pixels, and therefore, they require much computation time and are not suited for combining with a deep CNN.

In this paper, we propose a novel network architecture called *weight-shared multi-stage network* (WSMS-Net). In contrast to ordinary CNNs built by stacking a number of convolution layers, the WSMS-Net has a multistage architecture consisting of multiple ordinary CNNs arranged in parallel. A conceptual diagram of the WSMS-Net is depicted in Fig. 1. Each stage

R. Takahashi, T. Matsubara, and K. Uehara were with Graduate School of System informatics, Kobe University, 1-1 Rokko-dai, Nada, Kobe, Hyogo 657-8501, Japan. E-mails: takahashi@cs.ai.kobe-u.ac.jp, matsubara@phoenix.kobe-u.ac.jp, and uehara@kobe-u.ac.jp.

of the WSMS-Net consists of all or part of the same CNN: The weight of each convolution layer in each stage is shared with that of the corresponding layer of the other stages. Each stage is given a scaled image of an original input image. The features extracted at the all the stages are integrated at the *integration layer* and used for image classification. Thanks to this architecture, similar features are extracted even from objects of different scales at different stages, and thereby, the same objects of different scales are considered to be in the same class. We apply the WSMS-Net to existing deep CNNs and the experimental results demonstrate that the combination with the WSMS-Net improves the classification accuracy of the original deep CNNs by acquiring the scale invariance or at least the robustness to scaling of objects.

II. RELATED WORKS

A. Deep CNNs

The CNNs have been introduced with the shared weight parameters, the limited receptive fields, and the pooling layer [13], inspired by biological architecture of mammalian visual cortex [8], [1]. Thanks to the architecture, the CNNs suppress increase in the number of weight parameters and acquire a robustness to parallel shift of objects in image. In recent years, CNNs have made remarkable improvements in image classification with the deeper and deeper architecture. The CNNs with many convolution layers such as AlexNet [11], VGG [19], and GoogLeNet [20] have achieved high expression ability by increasing the number of internal weight parameters and classified images with high precision. However, by stacking more and more convolution layers in CNNs, gradient vanishing problem arises [18]. In general, CNNs use the back-propagation algorithm, which calculates the gradient obtained at the output layer, back-propagates the gradient in the direction from the output layer to the input layer, and updates the weight parameters. When the layers of the network become very deep, the gradient information from the output layer does not pass well to the input layer and layers near the input, and learning procedure does not progress well. Against this problem, ResNet [22] and DenseNet [6] were proposed as the new network architectures that enable the learning with further deep architecture. ResNet is the network that recorded the highest accuracy in ILSVRC image classification task in 2015. From 2015, following ResNet, many ideas and new architectures have been proposed [4], [23], [7], [21], [3], and in ILSVRC 2016, an extended version of ResNet broke the record [23]. The main idea of these network is addition of extra connections, which propagate the gradient information successfully and enable robust learning even with 100 or more convolution layers.

B. Deep CNNs against Vanishing Gradient

ResNet: ResNet (Residual Network) [22] is a new form of CNN that enables the CNN to overcome the gradient vanishing problem. A simple structure of ResNet is depicted in Fig. 2. The basic structure of ResNet is called *residual block*, which is composed of two convolution layers and a

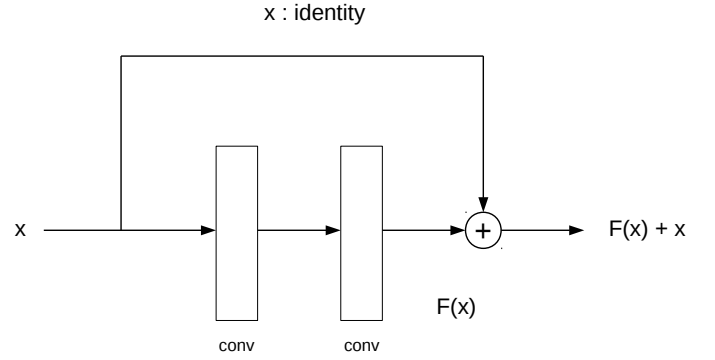


Fig. 2. A residual block of ResNet. x is a shortcut term performing identity mapping simply and $F(x)$ is the residual mapping through two convolution layers. These two mappings are just added and $F(x) + x$ is output of this residual block.

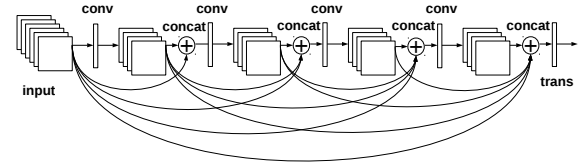


Fig. 3. A dense block of DenseNet. The *conv* denotes a convolution layer and the squared objects are feature maps just after the convolution layer. At *concat* point, all feature maps coming from the previous layers are concatenated in channel direction and the final *trans* layer is a cushion consisting of 1×1 convolution and pooling layers.

shortcut connection which does not pass through the convolution layers: The shortcut connection simply performs identity mapping. The result obtained from the usual convolution layers is denoted as $F(x)$ and that from the shortcut connection is denoted as x . The output obtained from the whole block is $F(x) + x$. In deep CNNs, the gradient information has a risk of vanishing at the feature extraction in the convolution layer $F(x)$. However, by adding the identity mapping here, the gradient information can be transmitted without any risk of vanishing. The whole network of ResNet is built by stacking this residual block in many layers. ResNet needs no extra parameters and the gradient can be calculated in the same manner as the conventional CNNs.

DenseNet: DenseNet (Densely Connected Convolutional Network) [6] is an improved network based on the concept of ResNet and its accuracy of image classification is better than that of ResNet. A basic architecture of DenseNet is depicted in Fig. 3. As its name implies, the structure of DenseNet connects layers densely. Differently from ResNet, in which a shortcut is connected across each two convolution layers, a shortcut is connected from one layer to all subsequent layers in DenseNet. In addition, the output of the shortcut is not added to but is concatenated with the output of the subsequent convolution layer in the direction of the channels of the feature map. As the depth of network is deepened, the number of the channels of the feature map increases linearly. This feature is considered to be better than the conventional architectures of ordinary CNNs and ResNet, with which the number of the channels is doubled at fixed timings like 16, 32, 64, and 128 [6]. The number k

of the increased channels per layer is called *growth rate*. This basic architecture depicted in Fig. 3 is called *dense block* and the whole network of DenseNet is built by stacking the dense block. The original DenseNet is constructed with three dense blocks: The size of the feature map in a single dense block is fixed and the shortcuts are not connected across each dense block. Instead, a 1×1 convolution layer that does not change the size or the number of channels is placed after a dense block for cushioning, followed by a pooling layer thinning out the features. A convolution layer is placed before the first dense block for increasing the number of channels of the input image.

Although the development of new network models such as ResNet and DenseNet has overcome the gradient vanishing problem, the CNNs still do not have established solution to invariance to geometrical transformations such as scaling and rotation. Lack of such invariance is an obstacle of improvement of deep CNNs.

C. Deep CNNs against Geometrical Transformations

Several studies have tried to absorb the geometrical transformations by neural networks [15], [2]. Spatial Transformer Network (STN) and Deep Recurrent Attentive Writer (DRAW) infer the parameters such as position and angle of the geometrical transformation of an object in image and correct the shape of the object by using neural networks. They are general and powerful approaches against geometrical transformations. However, the STN requires additional CNNs to localize and correct the object in image. DRAW requires repeated computation between all-to-all pairs of the input image and the output image to adjust parameters gradually for each image. They require many additional parameters and computation time, and thereby, they are not suited for combining with a deep CNN.

III. WEIGHT-SHARED MULTI-STAGE NETWORK

A. Architecture of Weight-Shared Multi-Stage Network

In this study, we propose a novel network architecture called *weight-shared multi-stage network* (WSMS-Net) to acquire the robustness to scaling of objects. The basic architecture of WSMS-Net is shown in Fig. 4. The WSMS-Net is constructed with S stages and each stage is just all or part of an ordinary CNN that can be divided into k convolution blocks for $k \geq S$. Each convolution block is constructed with some convolution layers. Between the blocks, pooling layer such as max pooling [17] is placed to downsize the feature map by half. The first stage is constructed with k blocks while the second stage has $(k - 1)$ blocks, the third stage has $(k - 2)$ blocks, and so on. The blocks at the same depth of all the stages completely share their weight parameters of convolution layers. Note that when batch normalization [9] is employed, its parameters are not shared. The first stage is given the original image, while the second stage is given the image resized to half by average pooling [13]. In a similar way, the s -th stage is given the image resized to half of the size of the image given to the $(s-1)$ -th stage. Therefore, the sizes of the output feature maps of all the stages are the same. Then, these feature maps are concatenated in the channel direction at the ends of

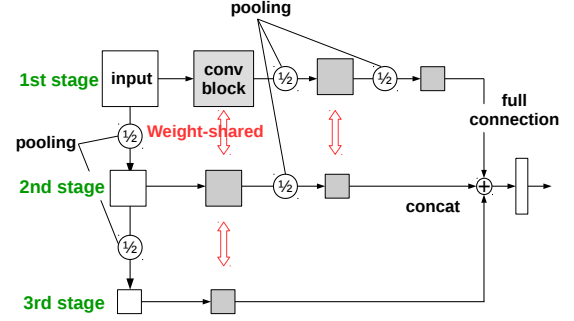


Fig. 4. A simple weight-shared multi-stage network (WSMS-Net) consisting of three stages. The input image is resized to half and quarter for the second stage and the third stage. Each *conv block* consists of some convolution layers and is followed by pooling layer, at which the sizes of the feature maps are reduced to half. *Weight-shared* indicates that the weight parameters of all the convolution layers in the same block in each stage are completely shared with each other. At *concat*, the outputs of all the stages are concatenated in the channel direction.

all the stages. After concatenation, all the feature maps are integrated at the integration layer and are used as input to the last fully connected layer for classification process. Thanks to these processes, various features are extracted from the input image of multiple sizes in the multiple stages and contribute to the classification.

B. Concepts of Weight-Shared Multi-Stage Network

We explain the detailed concept of the WSMS-Net. Let us consider the image A and the image B depicted in Fig. 1. There are two stages: The first stage is given the original $N \times N$ image as input and the second stage is given $\frac{N}{2} \times \frac{N}{2}$ image, being resized to half, as input. The image A is an image depicting the whole of an automobile and the image B is an image containing only the part around the left front tire of a sports car. In the training phase, the WSMS-Net is given the image A and learns the features related to the automobile. In the first stage, since the size of the image remains at $N \times N$, the WSMS-Net learns the local features of the car such as the shapes of tires, windows, headlights, and license plate. In the second stage, since the image is resized to $\frac{N}{2}$, the WSMS-Net learns the features such as the whole shape of the car but ignores the local features learned in the first stage. The image B is an image depicting only a part of the sports car; a tire and a headlight. When the WSMS-Net is given the image B, the feature maps corresponding to the local features learned with the image A in the first stage naturally respond to the image B in the second stage. An image of the whole of an object in a close-up and another image of the object in a long shot also share their features in the second stage and the first stage in a similar way.

We emphasize that the WSMS-Net is not a data augmentation [14]. With a data augmentation, the CNNs are always given a single image resized with various scaling factors and try to classify it without other clues. Therefore, the CNNs should learn the features of a great variety by using numerous weight parameters. On the other hand, the WSMS-Net is

required to find a feature contributing to classification in at least one of its multiple stages, which are given images of multiple scales. In Fig. 1, the WSMS-Net recognizes the object in the image B as a car according to the local features in the second stage. Then, the first stage is not required to learn more detailed features such as tire tread pattern. As a result, the variety of the features that the WSMS-Net should learn is reduced and the increase in the number of weight parameters is suppressed. The WSMS-Net can be considered to share weight parameters in front-back direction for images, while the ordinary CNNs share weight parameters only in vertical and horizontal directions.

C. Combination with Existing CNNs

By combining with the WSMS-Net, ResNet and DenseNet are expected to classify the images that could not be dealt with by the original CNNs. In this paper, WSMS-Net combined with ResNet is called *WSMS-ResNet* and WSMS-Net combined with DenseNet is called *WSMS-DenseNet*. For building WSMS-ResNet and WSMS-DenseNet, the number of stages and the shape of the integration layer should be optimized. The integration layer is an extra layer placed just after the concatenation layer that concatenates all the feature maps from all the stages, and integrates all the feature maps before the classification. For comparison, we also introduce a more trivial network called *multi-stage network* (MS-Net). MS-Net has multiple stages like WSMS-Net but each stage has its own parameters like an ensemble of multiple CNNs (e.g., [24]). MS-Net has much more parameters than WSMS-Net but does not have the feature of WSMS-Net described above.

IV. OPTIMIZATION OF WSMS-NET

A. Number of Stages

The number of stages is a fundamental factor of WSMS-Net. Both ResNet and DenseNet can be divided into several compartments by pooling layers (average pooling [13] or max pooling [17]), which downsize the feature maps by half. A compartment of ResNet consists of several residual blocks and a compartment of DenseNet consists of a single dense block. Thus, one compartment can be considered as a single convolution block of WSMS-Net, and WSMS-ResNet and WSMS-DenseNet have up to the number of the compartments for obtaining the feature maps of the same size at the integration layer.

B. Integration Layer

We have not yet described the integration layer in detail. In WSMS-Net, the feature maps obtained from all the stages are concatenated. The final number of the channels of the feature map is larger than that in the original network. The final feature map is often given to a global pooling layer and becomes a feature vector before the last fully connected layer, whose outputs denote the inferred class labels. Before the global pooling layer, the integration layer is given the concatenated feature map. In the simplest way, the integration layer does nothing, and the last fully connected layer is given the feature

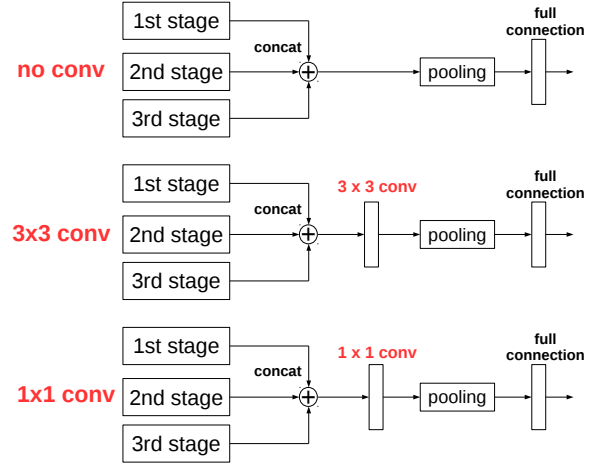


Fig. 5. Three types of integration layers. With *no cnn*, the concatenated feature map is directly given to the global pooling layer followed by the last fully connected layer. $3 \times 3 \text{ conv}$ and $1 \times 1 \text{ conv}$ are literally a 3×3 convolution layer and a 1×1 convolution layer, placed between the concatenated feature map and the global pooling layer.

vector longer than that of the original network. This integration layer is called *no conv*, hereafter. Otherwise, the integration layer can be a convolution layer with the $k \times k$ kernel and the c output channels. We evaluated a 3×3 convolution layer as the integration layer because both ResNet and DenseNet mainly employ 3×3 convolution layers. This integration layer is also called $3 \times 3 \text{ conv}$. In addition, we evaluated a 1×1 convolution layer as the integration layer called $1 \times 1 \text{ conv}$. We set the number c of the output feature maps to 128 in this study.

V. EXPERIMENTS

A. Classification of CIFAR-10 and CIFAR-100 by WSMS-ResNet

WSMS-ResNet for CIFAR-10 and CIFAR-100: This section evaluates ResNet [22] with WSMS-Net for CIFAR-10 and CIFAR-100 [10]. CIFAR-10 and CIFAR-100 are datasets of 32×32 RGB images of natural scene objects: Each consists of 50,000 training images and 10,000 test images. Each image is given one of 10 class labels by hand in CIFAR-10 dataset, while each image is given one of 100 class labels in CIFAR-100 dataset: The number of per class images is reduced in CIFAR-100. The 110-layer ResNet was reported as the model achieving the highest accuracy for CIFAR-10 dataset when the reference [22] was published. The 110-layer ResNet has three compartments and each compartment consists of 18 residual blocks. The sizes of the feature maps are 32×32 , 16×16 , and 8×8 in the first, second, and third compartments, respectively. The number of the channels of the feature map is three at the input and is increased to 16, 32, and 64 by the convolution layers placed before the first, second, and third compartments, respectively. After the third compartment, a global average pooling is performed on the 8×8 feature map of 64 channels, resulting in a 1×1 feature map of 64 channels, i.e., a 64-dimensional feature vector. This feature vector is given to the final fully connected layer for classification.

TABLE I
TEST ERROR RATES OF WSMS-RESNET AND ORIGINAL RESNETS ON CIFAR-10 AND CIFAR-100 DATASETS.

Network	depth	C10+		C100+	
		#params	Error (%)	#params	Error (%)
ResNet	110	1.73M	6.61	1.73M	28.77
ResNet	116	1.82M	6.60	1.83M	28.34
ResNet	122	1.92M	6.57	1.93M	28.39
MS-ResNet (1×1 conv)	111	2.23M	6.41	2.25M	27.23
WSMS-ResNet (no conv)	110	1.73M	7.23	1.74M	28.72
WSMS-ResNet (3×3 conv)	111	1.86M	6.41	1.87M	27.16
WSMS-ResNet (1×1 conv)	111	1.75M	6.36	1.76M	27.45

We constructed a WSMS-ResNet by combining the 110-layer ResNet [22] with WSMS-Net. The number of stages of the WSMS-ResNet was set to three so that each compartment of the 110-layer ResNet corresponds to a convolution block of the WSMS-Net. An input image having the size of 32×32 was downsized to 16×16 for the second stage, and was downsized to 8×8 for the third stage. The first stage was just the same as the original 110-layer ResNet before the global pooling layer. The second stage was composed of the first two compartments, in which the size of the feature map was 16×16 in the first convolution block and 8×8 in the second convolution block. Also, the third stage is composed of the first compartment of the original 110-layer ResNet using the 8×8 feature map. The integration layer was given the feature map of $64 + 32 + 16 = 112$ channels. With the integration layer of no conv, the feature maps became a 112-dimensional feature vector through the global average pooling. Otherwise, the feature map was projected to 8×8 feature maps of 128 channels through the integration layer of 3×3 conv or 1×1 conv before the global average pooling. The hyperparameters and other conditions of the WSMS-ResNet followed those used for the 110-layer ResNet in the original study [22]. All the images were normalized with mean and variance of each channel of the feature map, and 4×4 cropping and random flipping in the horizontal direction were employed as further data normalization and data augmentation. Batch normalization [9] and ReLU activation function [16] were used. The weight parameters were initialized following the algorithm proposed in the reference [5]. The WSMS-ResNet was trained using the momentum SGD algorithm with the momentum parameter of 0.9, the mini-batch size of 128, and the weight decay of 10^{-4} over 164 epochs. The learning rate was set to 0.01, and then, it was changed to 0.1, 0.01, and 0.001 at the 2nd, the 82nd, and the 123rd epochs, respectively.

Classification Results: Table I summarizes the results of the WSMS-ResNets and the original ResNets. Since the original study [22] did not report the results for CIFAR-100, we evaluated the 110-layer ResNet for CIFAR-100 in addition to the WSMS-ResNet for comparison. Following the original study [22], we obtained the mean test error rate of five trials. For CIFAR-10, the error rate of our proposed WSMS-ResNet with the 1×1 conv integration layer is 6.36 %: This accuracy is obviously superior to the error rate of 6.61 %

obtained from the original ResNet. However, the number of parameters of WSMS-ResNet was increased from 1.73M to 1.75M. For fair comparison, we also evaluated the deeper ResNets of 116-layer and 122-layer. In spite of a larger increase in the number of parameters, the deeper ResNets achieved the accuracies at a similar level to the 110-layer ResNet. These results demonstrate that the increase in the number of parameters along with the increase in the depth has a limitation in improvement of classification accuracy and that our proposed WSMS-Net enables the original ResNet to achieve a better classification accuracy. MS-ResNet achieved a better error rate than the original ResNet but the worse error rate than the WSMS-ResNet in spite of a massive increase in the number of parameters, indicating that the improvement of the WSMS-ResNet is thanks to the shared parameters rather than the network architecture. Even though the WSMS-ResNet has the limited increase in the number of parameters thanks to shared weight parameters compared with the MS-ResNet, the WSMS-ResNet and the MS-ResNet have the equal increase in the number of calculations owing to the second and third stages. Therefore, we evaluated the number of multiplications in all convolution layers in the original ResNet and the WSMS-ResNet. The number of multiplications is about 252M in the original 110-layer ResNet and is about 301M in the 111-layer WSMS-ResNet with the 1×1 conv integration layer: Our proposed WSMS-ResNet gets only a 20 % increase in the number of multiplications. In general, the second stage requires less than 25 % computations of the first stage since the second stage is given an image resized to half.

In the case of CIFAR-100, the WSMS-ResNet with 1×1 conv integration layer also surpassed the original ResNet. Unlike the case of CIFAR-10, the deeper ResNets and the MS-ResNet achieved better results than the original ResNet, and the WSMS-ResNet with 3×3 conv integration layer achieved the best result. We consider that the difference between CIFAR-10 and CIFAR-100s is caused by complexity of classification task. Classification of CIFAR-100 is more difficult because of the larger number of classes and the limited numbers of samples, and thus, requires much more weight parameters than classification of CIFAR-10. Therefore, the WSMS-ResNet with 3×3 conv integration layer, having more weight parameters, is simply more advantageous than the WSMS-ResNet with 1×1 conv integration layer. The deeper

TABLE II
TEST ERROR RATES OF WSMS-DENSENET AND ORIGINAL DENSENETS ON CIFAR-10 AND CIFAR-100 DATASETS.

Integration Type	growth rate k	C10+		C100+	
		#params	Error (%)	#params	Error (%)
DenseNet	24	27.2M	3.74	27.2M	19.25
DenseNet	26	31.9M	3.82	31.9M	18.94
MS-DenseNet (1×1 conv)	24	41.3M	4.18	41.3M	18.70
WSMS-DenseNet (no conv)	24	27.4M	4.54	27.8M	20.11
WSMS-DenseNet (3×3 conv)	24	32.7M	3.54	32.7M	19.16
WSMS-DenseNet (1×1 conv)	24	28.0M	3.51	28.0M	18.45

ResNet and the MS-ResNet are also better than the original ResNet thanks to their large numbers of weight parameters.

B. Classification of CIFAR-10 and CIFAR-100 by WSMS-DenseNet

WSMS-DenseNet for CIFAR-10 and CIFAR-100: In this section, we combined DenseNet of growth rate $k = 24$ with WSMS-Net, and constructed WSMS-DenseNet. The DenseNet ($k = 24$) was reported as the model achieving the second highest classification accuracy among the results of CIFAR-10 and CIFAR-100 datasets in the original study [6]. Note that we evaluated the model achieving the highest classification accuracy, DenseNet-BC, implemented on the source code published by the authors of the original study at <https://github.com/liuzhuang13/DenseNet>, but failed reproducing the results: This is why we used the DenseNet ($k = 24$). The DenseNet ($k = 24$) has three dense blocks, and the sizes of the feature map are 32×32 , 16×16 , and 8×8 in the first, second, and third dense blocks, respectively. The number of the channels is three at the input, is increased to 16 by the convolution layer placed before the first dense block, and is increased by the growth rate $k = 24$ at every convolution layer. Each dense block consists of 32 convolution layers. Therefore, the number of the channels reaches to $16 + 24 \times 32 \times 3 = 2320$. After the third dense block, a global average pooling is performed, resulting in a 2320-dimensional feature vector given to the last fully connected layer for classification.

We constructed WSMS-DenseNet of growth rate $k = 24$ by combining DenseNet with WSMS-Net. Each dense block of DenseNet was treated as a convolution block of WSMS-Net. The final numbers of the channels of the first, second, and third stages were $16 + 24 \times 32 \times 3 = 2320$, $16 + 24 \times 32 \times 2 = 1552$, and $16 + 24 \times 32 \times 1 = 784$, respectively. Hence, the integration layer was given the feature map of $2320 + 1552 + 784 = 4656$ channels. The hyperparameters and the other conditions of WSMS-DenseNet followed those of the original DenseNet ($k = 24$). All the images were normalized with mean and variance of each filter, and 4×4 cropping and random flipping in the horizontal direction were employed as further data normalization and data augmentation. Batch normalization [9] and ReLU activation function [16] were used. The weight parameters were initialized following the algorithm proposed in the reference [5]. WSMS-DenseNet was trained

using the momentum SGD algorithm with the momentum parameter of 0.9 and the weight decay of 10^{-4} over 300 epochs. The learning rate was set to 0.1, and then, it was reduced to 0.01 and 0.001 at the 150th and the 225th epochs, respectively. Note that only the mini-batch size was changed from 64 to 32 owing to the capacity of the computer we used.

Classification Results: Table II summarizes the results of WSMS-DenseNet ($k = 24$) and the original DenseNets. For CIFAR-10 dataset, the error rate of our proposed WSMS-DenseNet with the 1×1 conv integration layer is 3.51 %, which is better than the error rate of 3.74 % obtained from the original DenseNet ($k = 24$). We also evaluated DenseNet ($k = 26$) for fair comparison as is the case with WSMS-ResNet for CIFAR-10 dataset and confirmed that DenseNet ($k = 26$) achieved the worse error rate of 3.82 % in spite of a large increase in the number of weight parameters. The MS-DenseNet was also evaluated and achieved a worse error rate. The number of multiplications is about 6,889M in the original 100-layer DenseNet and is about 8,454M in the 100-layer WSMS-DenseNet with the 1×1 conv integration layer: Our proposed WSMS-DenseNet gets only a 20 % increase in the number of multiplications.

Also for CIFAR-100, our proposed WSMS-DenseNet with the 1×1 conv integration layer achieved better accuracy of 18.45 % than the original DenseNet. The DenseNet ($k = 26$) and the MS-DenseNet also achieved accuracies better than the original DenseNet ($k = 24$) but worse than the WSMS-DenseNet. In conclusion, regardless of the difference in the structure between ResNet and DenseNet, the experimental results demonstrate that the increase in the depth has a limitation in improvement of classification accuracy and the WSMS-Net with 1×1 conv integration layer achieved a better accuracy than the original networks.

Misclassified Images: We collected test images from CIFAR-10 that the DenseNet ($k = 24$) and the DenseNet ($k = 26$) misclassified but the WSMS-DenseNet ($k = 24$) classified correctly for evaluating our proposed WSMS-Net. Random examples of such test images are shown in Fig. 6. Many images in Fig. 6 show objects that were taken in close-up and whose large portions were cropped, or show objects taken against backgrounds in long shot. For comparison, example images were randomly chosen from all the test images as shown in Fig. 7. Comparison between them demonstrates that the WSMS-Net additionally classifies images showing scaled

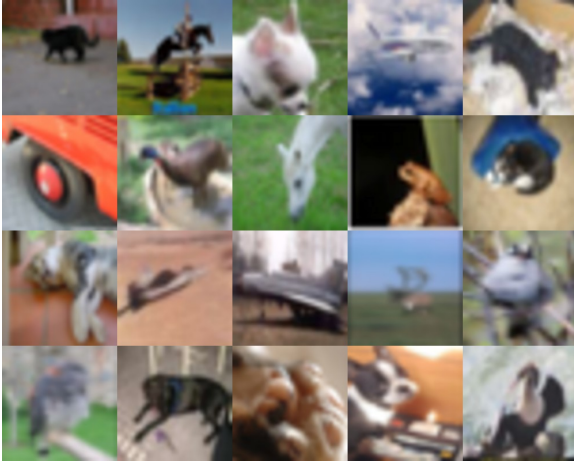


Fig. 6. Examples of CIFAR-10 test images misclassified by the DenseNet ($k = 24$) and the DenseNet ($k = 26$) but classified by the WSMS-DenseNet ($k = 24$, 1×1 conv) correctly.

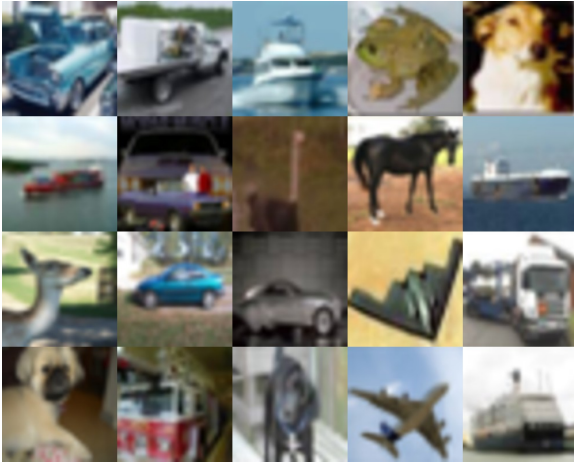


Fig. 7. Examples of CIFAR-10 test images.

objects. We conclude that the WSMS-Net acquires the scale invariance or at least is more robust to scaling of objects, and therefore, it achieves better classification accuracy than the original CNNs in spite of the limited increase in the number of weight parameters. A more detailed comparison is a future work.

VI. CONCLUSION

In this study, we propose a novel network architecture of convolutional neural networks (CNNs) called *weight-shared multi-stage network* (WSMS-Net) to improve classification accuracy by acquiring the robustness to scaling of objects. The WSMS-Net consists of multiple stages of CNNs given input images of different sizes. All the feature maps obtained from all the stages are concatenated and integrated at the ends of the stages. The experimental results demonstrated that the WSMS-Net achieved better classification accuracy and had higher robustness to scaling of objects than existing CNN model. Future work include a more detailed evaluation

of the scale invariance, evaluation with another dataset, and evaluation with another architecture of CNN.

REFERENCES

- [1] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.
- [2] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra, "DRAW: A Recurrent Neural Network For Image Generation," *arXiv*, pp. 1–11, 2015.
- [3] D. Han, J. Kim, and J. Kim, "Deep Pyramidal Residual Networks," *arXiv*, pp. 1–9, 2016.
- [4] K. He and X. Zhang, "Identity mappings in deep residual networks," *Lecture Notes in Computer Science*, vol. 9908 LNCS, no. 1, pp. 630–645, 2016.
- [5] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. of the IEEE International Conference on Computer Vision (ICCV2016)*, vol. 11-18-Dece, 2016, pp. 1026–1034.
- [6] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *arXiv*, pp. 1–12, 2016.
- [7] K. Q. Huang, Gao and Sun, Yu and Liu, Zhuang and Sedra, Daniel and Weinberger, "Deep Networks with Stochastic Depth," in *Proc. of European Conference on Computer Vision (ECCV2016)*, vol. 9905, 2016, pp. 646–661.
- [8] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *The Journal of Physiology*, vol. 195, no. 1, pp. 215–43, 1968.
- [9] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv*, pp. 1–11, 2015.
- [10] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," *Technical report, University of Toronto*, pp. 1–60, 2009.
- [11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proc. of Advances In Neural Information Processing Systems (NIPS2012)*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.
- [12] Q. Le, J. Ngiam, Z. Chen, D. H. Chia, and P. Koh, "Tiled convolutional neural networks," in *Proc. of Advances In Neural Information Processing Systems (NIPS2010)*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 1279–1287.
- [13] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.
- [14] C. Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Proc. of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS2015)*, vol. 2, 2015, p. 5.
- [15] J. Max, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial Transformer Networks," in *Proc. of Advances In Neural Information Processing Systems (NIPS2015)*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2017–2025.
- [16] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proc. of the 27th International Conference on Machine Learning (ICML2010)*, no. 3, 2010, pp. 807–814.
- [17] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," in *Nature Neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.
- [18] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.
- [19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *Proc. of International Conference on Learning Representations (ICLR2015)*, 2015, pp. 1–14.
- [20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR2015)*, vol. 07-12-June, 2015, pp. 1–9.
- [21] S. Targ, D. Almeida, and K. Lyman, "ResNet In ResNet: Generalizing Residual Architectures," in *Workshop on International Conference on Learning Representations (ICLR2016)*, no. 1, 2016, pp. 1–4.
- [22] S. Wu, S. Zhong, and Y. Liu, "Deep residual learning for image steganalysis," *Multimedia Tools and Applications*, pp. 1–9, 2017.

- [23] S. Zagoruyko and N. Komodakis, “Wide Residual Networks,” *arXiv*, pp. 1–15, 2016.
- [24] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, “Single-Image Crowd Counting via Multi-Column Convolutional Neural Network,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR2016)*, 2016, pp. 589–597.